The Power Of Two

Musa Al-hassy

McMaster University
alhassm@mcmaster.ca

January 18, 2017

Plato and friends

Are mathematical objects real? Explain the concept of "two" without using the concept of "number"!

Who is Boole?

Definition

The Booleans $\mathbb{B}$ is a set of two, and only two, elements denoted *true* and *false*.

"two, and only two" means we have

Decomposition / Pattern Matching / $\mathbb{B}$-Induction

every Boolean p is either *true* or *false*.

BNF Grammars —recall second year CS?

More generally,

Declaring a new type whose elements have *n*-possible "shapes"

mytype ::= $\text{construction}_1 \mid \cdots \mid \text{construction}_n$

Such a declaration means that (we claim) there is a type `mytype` and it is the smallest type with these constructions:

Decomposition / Pattern Matching / mytype-Induction

every element of `mytype` is uniquely of the shape constructor_i, for some (unique!) i and for some variables needed in the construction.

(Programming language **Haskell** supports this approach to data-types!)

An Example BNF Grammar —or specification vs representation

Even digits and even numbers

$\text{evenDigit} ::= 0 \mid 2 \mid 4 \mid 6 \mid 8$ $\qquad$ $\text{Evens} ::= \mathbb{N}\ \text{evenDigit}$

- What do these declarations claim?
- What is the induction principle for the first type?
- The induction principle for the second says that an element of `Evens` is uniquely of the shape: "some natural number followed by some even-digit".

These are claims. How do we realize/implement these types if we really wanted to?

Know at least two ways! Imperative and dependently-typed!

What is a `construction`?

Extremely important type

$$t ::= c \mid x \mid f(t_1, \ldots, t_n)$$

Term ::= constant | variable | application to other terms

Exercise: define the types needed in the definition of `Term`!

Important subtlety: f above is a function symbol!

Think, class-object constructor method!

Unique Equality

The Booleans $\mathbb{B}$ have an equality denoted $- \equiv -$. Besides the equivalence relation properties —what are they?—, it is characterised by the axiom

$$(p \equiv q) \equiv r \quad = \quad p \equiv (q \equiv r)$$

- Why so special and not use traditional equality symbol '$=$'?
- Difference between '=' and '≡?' Conjunctive vs Associative!
- How is this written in ACSL and traditional maths?

What's wrong with the phrase "$x = (y = z)$" for numbers? How does language C interpret this?

Identity of Equivalence

The reflexitivity axiom for Booleans says

$$(p \equiv p) = true$$

But '$=$' and '$\equiv$' are synonyms for the same concept but with different conventions, and that '$\equiv$' is associative gives us our first theorem:

Right Identity of Equivalence

$$p \equiv (p \equiv true)$$

But the symmetry axiom then gives us,

Left Identity of Equivalence

$$p \equiv (true \equiv p)$$

Putting order into our lives

Numbers are ordered and so nice to work with, what about the Booleans?

Implication

The Booleans $\mathbb{B}$ have a partial order —recall Sheet2!— denoted $- \Rightarrow -$. Besides the partial order properties, it is characterised by the axiom

$$\mathit{false} \Rightarrow \mathit{true}$$

That is, $\mathbb{B}$ is an ordered set of only two items where the smaller is called *false* and the larger is called *true*.

Compare with "$x \leq y$" on numbers!

Bounds

Since *false* is the least Boolean and *true* is the largest Boolean, we already have a theorem about all Booleans p:

$$\text{Left-Zero of Implication: } (\mathit{false} \Rightarrow p) \equiv \mathit{true}$$

$$\text{Right-Zero of Implication: } (p \Rightarrow \mathit{true}) \equiv \mathit{true}$$

Using the identity laws for equivalence, these can be simplified to

$$\text{Bottom of } \mathbb{B}\text{: } \mathit{false} \Rightarrow p$$

(ex-falso quodlibet or, "from false follows anything")

$$\text{Top of } \mathbb{B}\text{: } p \Rightarrow \mathit{true}$$

Compare with the extended-numbers: $-\infty \leq x \leq +\infty$.

Precarious Protocols

Warning!

The antisymmetry property reduces to

$$\text{if} \quad p \Rightarrow q \quad \text{and} \quad q \Rightarrow p \quad \text{then} \quad p \equiv q$$

For this reason, some write ‘ $\Longleftrightarrow$ ’ in-place of ‘$\equiv$’, but unfortunately that name implicitly suggests proving both implications to get at an equality(!) and this is seldom a good idea!

Compare with “$x \leq y$” on numbers! You don’t prove an equality with two containments!?!

let's avoid casing to limit complexity

Numbers are totally ordered, dude(tte), and as such have operations for minimum $\uparrow$ and maximum $\downarrow$.

Usual definition —case analysis

$$x \uparrow y := \text{if } x \leq y \text{ then } x \text{ else } y \text{ fi}$$

This is a good implementation, direct definition, but requires cases whenever we work with it!

Better characterisation —calculation friendly!

$$x \uparrow y \leq z \equiv x \leq z \wedge y \leq z$$

"$x \uparrow y$ is the *least upper bound* of both x and y"

where "$\wedge$" is read "and"

(remember first-year calculus? **Supremum?**)

Max and Min for $\mathbb{B}$

The maximum operation for the Booleans is actually denoted '$\vee$' and so the previous characterisation becomes —along with '$\wedge$' for min—

Disjunction/Max/$\vee$ and Conjunction/Min/$\wedge$

$$p \vee q \Rightarrow r \;\equiv\; (p \Rightarrow r) \wedge (q \Rightarrow r)$$

$$r \Rightarrow p \wedge q \;\equiv\; (r \Rightarrow p) \wedge (r \Rightarrow q)$$

The second one reads:
"r implies p and q" precisely when "r implies p, and r implies q"

Bits! One possible representation

Recall that a declaration such as

Boolean Expressions (BE)

$$\mathbb{B} \ ::= \ \mathit{true} \mid \mathit{false}$$

$$BE \ ::= \ \mathbb{B} \mid BE \equiv BE \mid BE \Rightarrow BE \mid BE \wedge BE \mid BE \vee BE \mid \neg BE$$

is a claim and so needs a "proof of concept" eventually,

Not the best, and we wont use this interpretation explicitly

- $\mathbb{B} := \{0, 1\}$, also known as $\mathbb{Z}_2$
- Equivalence is just usual equality on numbers '='
- Implication is just usual inclusion on numbers '$\leq$'
- Conjunction is just usual minimum on numbers ' $\downarrow$ '
- Disjunction is just usual maximum on numbers ' $\uparrow$ '
- Negation is "2's complement" or "subtraction from 1"

It is clear that this implementation satisfies the required axioms.
Next time, we'll discuss/formalise the axiomatic approach and use that!

References

The associativity of equivalence and the Towers of Hanoi problem
http://www.cs.nott.ac.uk/~psarb2/papers/abstract.html#Hanoi
"[...] greater use should be made of the associativity of equivalence. This note shows how the property is used in specifying the rotation of the disks in the well-known Towers of Hanoi problem. " –from the paper's abstract